[illegible]

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/888,955	06/25/2001	Calvin White	40101/02201	1402

30636 7590 11/05/2003

FAY KAPLUN & MARCIN, LLP
150 BROADWAY, SUITE 702
NEW YORK, NY 10038

EXAMINER	
KENDALL, CHUCK O	
ART UNIT	**PAPER NUMBER**
2122	3

DATE MAILED: 11/05/2003

Please find below and/or attached an Office communication concerning this application or proceeding.

PTO-90C (Rev. 10/03)

ARS

Office Action Summary	Application No.	Applicant(s)	
	09/888,955	WHITE, CALVIN	
	Examiner	Art Unit	
	Chuck O Kendall	2122	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on 25 June 2001 .

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) 1-28 is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) 1-28 is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

11)☐ The proposed drawing correction filed on _____ is: a)☐ approved b)☐ disapproved by the Examiner.

If approved, corrected drawings are required in reply to this Office action.

12)☐ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. §§ 119 and 120

13)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____ .

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

14)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application).

a) ☐ The translation of the foreign language provisional application has been received.

15)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)	4) ☐ Interview Summary (PTO-413) Paper No(s). _____ .
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)	5) ☐ Notice of Informal Patent Application (PTO-152)
3) ☐ Information Disclosure Statement(s) (PTO-1449) Paper No(s) _____ .	6) ☐ Other: .

DETAILED ACTION

1. This action is in response to the application filed 06/25/01.

2. Claims 1-28 have been examined.

Claim Rejections - 35 USC § 103

3. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

4. Claims 1-28 are rejected under 35 U.S.C. 103(3) as being unpatentable over Fresko et al. USPN 5,966,702 (hereinafter Fresko), in view of Merrick et al. USPN 6,339,841 B1 (hereinafter Merrick).

Regarding claims 1 and 12, Fresko discloses a system, comprising: a receiving module to receive a request to load a component (Col. 7 line 1 - line 10); a stack to record the request (10:61-64), a loader to fulfill the request (9:29), when the loading of the component is unsuccessful, contents of the stack are made available to a user to indicate the unsuccessfully loaded component (10:61-64, see partial results). Fresko doesn't explicitly disclose wherein when the request has been fulfilled the request is removed from the stack. However, Merrick does disclose this feature in a similar configuration (3:5-8). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to combine Fresko and Merrick because, using stacks in a Java environment is a general practice in the art and makes monitoring and managing objects within memory more efficient

Regarding claims 2 and 14, the system according to claim 1, further comprising:

an execution module to execute instructions contained in a component object, wherein the component object is created from information in the component (Fresko, 7:10).

Regarding claim 3, the system according to claim 1, wherein the receiving module, the stack and the loader reside on a development platform (Fresko, 7:1-15, see item# 200).

Regarding claim 4, the system according to claim 1, wherein the receiving module, the stack and the loader reside on an embedded device (Fresko, 7:1-15, see item# 200).

Regarding claim 5, the system according to claim 4, wherein the component resides external to the embedded device (fig 2, see 222, 223).

Regarding claim 6, the system according to claim 1, wherein the loader includes a plurality of loading modules in a hierarchical relationship and the component is loaded by a highest level loading module capable of loading the component (Fresko,47:40-50 for hierarchy see subclasses of ClassLoader).

Regarding claim 7, the system according to claim 1, wherein the stack contents are made available to the user via one of an on-screen display, a printout and a file (38:45-50).

Regarding claims 8 & 17, Fresko discloses a method of loading, comprising the steps of: receiving a request to load a first software module (Col. 7 line 1 - line 10); placing a representation of the first software module onto a stack (10:60-65, and 39:15); determining if the first software module is dependent on a second software module (Fresko, 10:35-39, see Multi class sets and required classes, [the secondary art also shows this limitation which is pretty common in the art during class packaing and loading, see Merrick, 1:55-60]); placing, when the first software module is dependent on the second software module, a representation of the second software module onto the stack (10:60-65); loading the second software module (Fresko, 10:35-39, see Multi class sets for second program), and loading the first software module (See definition for Multi class set, which by default loads atleast one class). Fresko doesn't explicitly disclose removing the representation of the first and second software module from the stack when the second software module has been successfully loaded. However, Merrick does disclose this feature in a similar configuration (3:5-8). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to combine Fresko and Merrick because, using stacks in a Java environment is a general practice in the art and makes monitoring and managing objects within memory more efficient.

Regarding claims 9 & 18, the method according to claim 8, further comprising the step of: making contents of the stack available to a user when the loading of one of the first software module and the second software module has been unsuccessful (Merrick,3:5-8).

Regarding claim 10, Examiner is applying the same rationale to claim, which is the method version (for method see, Fresko, 51:1-40) of the system claim as discussed in claim 7 above.

Regarding claim 11, the method according to claim 8, wherein the representation of the first software module is the name of the first software module (8:13, see class names).

Regarding claim 13, the system according to claim 12, wherein contents of the stack are made available to a user when fulfillment of the load request has been unsuccessful (Merrick, 3:5-8).

Regarding claim 15, the system according to claim 12, wherein, when the software component in the load request is dependent on an additional software component, an additional load request for the additional software component is received (Fresko, 10:35-39, see Multi class sets and required classes, [the secondary art also shows this limitation which is pretty common in the art during class packaing and loading, see Merrick, 1:55-60]) by the loader and the loader pushes a representation of the additional software component onto the stack when the additional load request is received and pops the representation of the additional software component off of the stack when the additional load request has been successfully fulfilled (Merrick, 3:5-8, Examiner deems this to be inactive when it has been fulfilled).

Regarding claim 16, the system according to claim 15, wherein contents of the stack are made available to a user when fulfillment of the additional load request has been unsuccessful and the contents of the stack include the representation of the software component and the additional software component (Merrick, 3:5-8).

Regarding claim 19, Examiner is applying the same rationale to claim, which is the method version (for method see, Fresko, 51:1-40) of the system claim as discussed in claim 7 above.

Regarding claim 20, the method according to claim 17, wherein the loading steps, the placing steps and the removing steps are performed by a Java class loader (Fresko, 44:4).

Regarding claim 21, the method according to claim 17, further comprising the steps of:

creating a first Java class object from the loaded first Java class (Fresko, 40:34); and executing instructions included in the first Java class object (Fresko, 44:10-12).

Regarding claim 22, the method according to claim 21, wherein the executing step is performed by a Java Virtual Machine (Fresko, 44:10-12).

Regarding claim 23, Fresko discloses a system, comprising: a stack to record a load request for a Java class (Col. 10, lines 61-64); a Java class loader to receive and fulfill the load request for the Java class (Col. 7 line 1 - line 10), a Java Virtual Machine to execute instructions contained in a Java class object, wherein the Java class object is created from information in the Java class (44:10-12). Fresko doesn't explicitly disclose wherein when the request has been fulfilled the request is removed from the stack. However, Merrick does disclose this feature in a similar configuration (3:5-8). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to combine Fresko and Merrick because, using stacks in a Java environment is a general practice in the art and makes monitoring and managing objects within memory more efficient.

Regarding claim 24, the system according to claim 23, wherein the Java class loader is one of a custom class loader and a default loader (Fresko, 9:28).

Regarding claim 25, the system according to claim 23, wherein contents of the stack are made available to a user when fulfillment of the load request has been unsuccessful (Merrick, 3:5-8).

Regarding claim 26, the system according to claim 23, wherein the stack, the Java class loader and the Java Virtual Machine reside on a development platform (fig 1, 100,102).

Regarding claim 27, the system according to claim 1, wherein the stack, the Java class loader and the Java Virtual Machine reside on an embedded device (fig 1, 102).

Regarding claim 28, the system according to claim 23, wherein the load request is received via a uniform resource locator (Fresko, 3:47-53).

Correspondence Information

6. Any inquires concerning this communication or earlier communications from the examiner should be directed to Chuck O. Kendall who may be reached via telephone at (703) 308-6608. The examiner can normally be reached Monday through Friday between 8:00 A.M. and 5:00 P.M. est.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Dam *can be* reached at *(703) 305-4552.*

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the Group receptionist whose telephone number is (703) 305-3900.

For facsimile (fax) send to 703-7467239 official and 703-7467240 draft

Chuck O. Kendall

Software Engineer Patent Examiner

United States Department of

[illegible]
WEI ZHEN
primary patent Examiner